In order to detect crosslink failures, STAR bridges on both ends of each crosslink exchange Hello SBPDUs periodically. These Hello SBPDUs are not forwarded by the STAR bridges to their neighbors. Each STAR bridge uses a predetermined timer for each of its crosslink neighbors to time out pending Hello SBPDUs. If a STAR bridge receives a Hello SBPDU from a STAR neighbor over a crosslink before an appropriate timer expires, the STAR bridge resets the timer. Otherwise, the STAR bridge assumes that the crosslink has failed and then transits to the Tree Learned state. In addition, the STAR bridge also multicasts a Distance vector change Notification SBPDU over the IEEE 802.1D spanning tree to all STAR bridges.

When a crosslink is recovered from a recent failure or a new crosslink link is enabled, the STAR bridges on both ends of the crosslink will first check to determine if the crosslink is eligible for supporting an enhanced forwarding path based on information in their respective BF Tables. If the crosslink is eligible, the STAR bridges will each independently multicast a DVCN_SBPDU over the IEEE 802.1D spanning tree to all STAR bridges.

Upon receiving a DVCN_SBPDU, a STAR bridge forwards it to all of its tree neighbors, and transits to the Tree Learned state. The STAR bridge makes use of a timer to remember the identity of the affected crosslink for a predetermined time-out period. When a DVCN_SBPDU identifying a given crosslink is received by a STAR bridge, and if no other DVCN_SBPDU identifying the same crosslink has been received within a current time-out period, the STAR bridge resets the timer and forwards the SBPDU to all its tree neighbors. Otherwise, the SBPDU is dropped.

I.G. Bridge Operation

As mentioned earlier, there are three kinds of MAC frames a STAR bridge would receive in this protocol: BPDU frames, SBPDU frames, and data frames. FIG. 5 is the STAR bridge operation flow-chart. When a MAC frame is received, it invokes different procedures for different kinds of frame.

BPDU frames received are processed by a procedure BPDU_Proc. FIG. 6 depicts the flow-chart for BPDU_Proc. When a Topology Change Notification BPDU is received, the bridge has to deactivate the non-tree ports selected by the path finding process and invalidate the entries of its ESL table. After that, standard BPDU processing is executed. In FIG. 6, Std_BPDU_Proc refers to the standard BPDU processing procedure, shown in FIG. 21. As the standard BPDU processing procedure can be found in the IEEE 802.1D standard which is incorporated herein by reference, the details of that procedure have been omitted for purposes of brevity.

There are four kinds of SBPDU frames as described in the Protocol Data Units Section. There is a procedure for processing each kind of SBPDU frames. DVCN_SBPDU_Proc procedure is for processing DVCN_SBPDU frames, and the flow-chart for this procedure is shown in FIG. 7. The details of DVC_SBPDU_Proc procedure for processing DVC_SBPDU frames and SLA_SBPDU_Proc

procedure for processing SLA_SBPDU frames will be discussed hereinafter with reference to Section V and Section IV respectively. The flow-charts for these procedures are shown in FIG. 12 and FIG. 9 respectively. A STAR bridge invokes a Data_Frame_Proc procedure when a data frame is received. The flow-chart for Data_Frame_Proc is shown in FIG. 8. Different procedures are executed depending on the current state of the bridge. The processing of data frames will be discussed further in Section IV and Section V.

II. Model

In this section, we describe the mathematical model we use in this invention. We will also define the notations for the proposed protocol. A summary can be found in Table A in the Appendix.

In the present invention the bridged LAN is represented as an undirected graph $G = (V, E)$ where V is the set of all bridges and E is the set of links connecting the bridges. Each link $(x, y) \in E$ is assumed to have a non-negative cost $c(x, y)$. For convenience, we let $c(x, y) = \infty$ if $(x, y) \notin E$. If there are several links between bridge x and bridge y, $c(x, y)$ should be the minimum among the costs of the links. A path in G is a loop-free tandem concatenation of links in E. The length of a path is the sum of the costs of all the links along the path. The distance between a pair of nodes, x and y, is the length of a shortest path connecting the nodes.

Bridge x is a *direct neighbor* of bridge y, and vice versa, if $(x, y) \in E$. $T = (V, E_T)$ is a tree subgraph of G representing an RST, wherein $(x, y) \in E_T$ if and only if (x, y) is an activated link in the RST. The links in E_T are referred to as *tree links* and the links in $E \setminus E_T$ as *non-tree links*. If $(x, y) \in E_T$, x and y are *tree neighbors*. A path in T is a *tree path*. A tree path originating at bridge s and terminating at bridge t is denoted *treepath(s, t)*. The distance of this tree path is denoted $d_T(s, t)$. Note that $d_T(x, y) = c(x, y)$ if x and y are tree neighbors. We refer to *Treepath(s, t)* are referred to as an *old bridge tree path* if it has at least one intermediate bridge (i.e., one other than the source and destination bridges) and every intermediate bridge on the path is an old bridge. B represents the set of STAR bridges. If s and t are STAR bridges, that is, $s, t \in B$, and there is an old bridge tree path between them, s is a *distant STAR neighbor* of t, and vice versa. If, in addition, s is an ancestor of t, then s is a unique *distant STAR ancestor neighbor* of t. Henceforth, the distant STAR ancestor neighbor of a bridge t will be referred to by *dsan(t)*. The set of distant STAR neighbors of n is represented by $N'_B(n)$.

The nearest common ancestor of x and y is the highest-level bridge on a tree path between x and y. If x is an ancestor of y, then x is necessarily the nearest common ancestor of x and y. Let the nearest common ancestor of x and y be denoted $nca(x, y)$. We say x and y are on different branches if $nca(x, y) \neq x$ and $nca(x, y) \neq y$. We call $(x, y) \in E \setminus E_T$ a *crosslink* if x and y are on different branches. x and y are *crosslink neighbors* then. FIG. 10 is an example of an undirected graph of a bridged LAN. Node r is the root. The solid lines are links in E_T and the dotted lines are non-tree links. Link (u, q) is a non-tree link but not a crosslink while (w, y), (u', v) and (v', z) are all crosslinks. Therefore, u and q are direct neighbors but neither tree neighbors nor crosslink neighbors. They are distant STAR neighbors though

since the path $u \rightarrow y \rightarrow q$ is an old bridge tree path. Table 1 summarizes the definitions of different kinds of neighbors.

Neighbor Type	Definition
Direct neighbors	$x, y \in V, (x, y) \in E$
Tree neighbors	$x, y \in V, (x, y) \in E_T$
Crosslink neighbors	$x, y \in V, (x, y) \in E \backslash E_T$ and $nca(x, y) \neq x$ and $nca(x, y) \neq y$
Direct STAR neighbors	$x, y \in B, (x, y) \in E$
Distant STAR neighbors	$x, y \in B$, *treepath(x, y)* is an old bridge tree path

Table 1: Neighbor Types

Since an old bridge sets the port associated with each of its non-tree links to blocking state, there is no way to use any such a link to forward data frames, even though the other side of the link is a STAR bridge. Therefore, a non-tree link may be used only if it connects two STAR bridges. Even so, such a link may not support any shortest path. There is yet another reason that a non-tree link is useless. Consider FIG. 10 where both bridges u and q are STAR bridges. This non-tree link between them is useless because the distance of this link must be larger than the tree path from u to q; otherwise, the spanning tree algorithm would have set q to be a child of u. In FIG. 10, *(u, q)* and *(w, y)* are ineligible links. A non-tree link that is obviously useless, as just described, for supporting any shortest path is termed an *ineligible link*. A non-tree link is termed *eligible* otherwise. In FIG. 10, *(u', v)* and *(v', z)* are eligible crosslinks.

A STAR *bridge graph* is defined as $G_B=(B, E_B)$ where B is the set of STAR bridges and $(x, y) \in E_B$ if and only if x and y are STAR neighbors, either direct or distant. $c'(x, y)$, the cost of link (x, y), is defined by the following formula.

$$c'(x, y) = \begin{cases} c(x, y) & \text{if } x \text{ and } y \text{ are direct but not distant STAR } neighbors \\ min(d_T(x, y), c(x, y)) & \text{if } x \text{ and } y \text{ are both direct and distant STAR neighbors} \\ d_T(x, y) & \text{if } x \text{ and } y \text{ are distant but not direct STARneighbors} \end{cases}$$

The STAR bridge graph of FIG. 10 is shown in FIG. 11.

III. Path Finding Process

The goal of this process is to compute the BF Table. In the best case, the BF Table has next hop and forwarding port information associated with a shortest path to a STAR bridge. The STAR bridge graph contains all tree paths among STAR bridges and all eligible non-tree links in the original bridged LAN. Therefore, the shortest path in G_B between a pair of STAR bridges x and y would be the best path which can be achieved in the bridged LAN after pruning ineligible links. Ideally, if every $c'(x, y)$ can be computed correctly, each STAR bridge can compute its own BF Table based on distance vectors.